applied sciences

MDPI

Article

Benchmarking of Multi-Class Algorithms for Classifying Documents Related to Stunting

Retno Kusumaningrum [1,*], Titan A. Indihatmoko [1], Saesarinda R. Juwita [1], Alfi F. Hanifah [1], Khadijah Khadijah [1] and Bayu Surarso [2]

1 Department of Informatics, Universitas Diponegoro, Semarang 50275, Indonesia; titanabrian@student.undip.ac.id (T.A.I.); saesarinda@if.undip.ac.id (S.R.J.); alfifauzia@student.undip.ac.id (A.F.H.); khadijah@live.undip.ac.id (K.K.)

2 Department of Mathematics, Universitas Diponegoro, Semarang 50275, Indonesia; bayus@lecturer.undip.ac.id

* Correspondence: retno@live.undip.ac.id; Tel.: +62-812-985-53236

Received: 27 September 2020; Accepted: 29 November 2020; Published: 2 December 2020

Abstract: Stunting is a condition in which children experience impaired growth and development, caused by malnutrition, repeated infections, and inadequate psychosocial stimulation. It often remains unrecognized due to a lack of awareness in the community. Therefore, the first step towards developing a solution for stunting is to understand the level of awareness and the sentiment of the community towards issues related to stunting. As online media are widely used in everyday life, they offer significant potential towards providing such an understanding. However, exploiting this potential requires extensive identification of documents containing discussions of stunting among lay people, to accurately gauge the awareness and sentiments of the community towards stunting. This task is a multi-class classification problem. We perform a benchmark study, using data from the Indonesian context, to comparatively evaluate the performances of four algorithms, i.e., logistic regression, naive Bayes, random forest, and support vector machine (SVM), and three extracted features, namely term occurrence, term presence, and term frequency-inverse document frequency (TF-IDF). The SVM method coupled with TF-IDF produced the highest accuracy value of 0.98, with a standard deviation of 0.03, due to its capability to automatically model the interaction between features.

Keywords: stunting; logistic regression; naive Bayes; random forest; support vector machine

1. Introduction

Stunting is a condition that affects children and is characterized by impaired growth and development, leading to their height being much lower than that of a healthy child of their age. It is caused by malnutrition, repeated infections, and inadequate psychosocial stimulation. Formally, a child is identified as being affected by stunting when their height is more than two standard deviations below the World Health Organization (WHO) Child Growth Standards median [1]. Stunting is estimated to affect a significant number of children worldwide, and its effects on affected individuals and communities are serious and long-term.

According to information obtained from WHO data, Indonesia ranks fifth highest among all countries in the number of people affected by stunting. Consequently, President Joko Widodo (Jokowi) of the Republic of Indonesia mentioned that reducing the incidence of stunting in the country is a key concern, and instructed the ministers and his staff at a press conference to develop an integrated action plan to reduce stunting in Indonesia [2]. Nevertheless, to reduce stunting rates in Indonesia, the development of an integrated action plan must be supported with concrete steps to understand and address the root causes of stunting in the country.

A significant factor that enables the prevalence of stunting is the lack of awareness around issues related to stunting, its causes, and its dangers. Commonly, stunting often advances unrecognized, particularly in communities where a short stature is considered normal [3]. Therefore, there is a need to understand the level of awareness and appreciation from the public regarding the causes and dangers of stunting. Such an understanding can enable stunting campaigns to be better targeted and more effectively reduce stunting rates in Indonesia.

In this study, we contribute to the development of such an understanding of community awareness and sentiment towards stunting issues, with a focus on the Indonesian context, by exploring the mining of public discussions and articles regarding stunting in Indonesia. As social media and other online media, including online news and crowd-sourced content, are used widely in everyday life, they offer significant potential for mining information about the level of awareness and the sentiment of the society towards stunting in Indonesia.

However, data on social and other online media can pertain to various subjects, and it can be challenging to retrieve data that is specifically related to stunting. Specifically, a document, e.g., a post or an article, might be unrelated to stunting, might contain information about stunting, or might contain a discussion of stunting among lay people. In order to correctly gauge the level of awareness and the sentiment of the community towards stunting, we need to extensively identify documents containing discussions related to stunting among lay people. Since there are three classes of documents as mentioned above, the task of automatically detecting a document containing a discussion of stunting among lay people is a multi-class classification task.

In this study, we perform a comparative evaluation of the performances of several algorithms, to provide a benchmark of algorithms for multi-class classification, and specifically for Indonesian language documents related to stunting issues. Such a benchmark can enable researchers and policy makers to choose the best-suited approach for extensively retrieving relevant discussions to study community awareness and sentiment towards stunting.

The rest of this paper is organized as follows. Section 2 describes prior studies on multi-class classification tasks for textual domain data, while the methodology adopted in our study is described in detail in Section 3. Section 4 discusses the experimental setup, and the experimental results and analysis. Finally, Section 5 concludes the paper.

2. Related Works

A multi-class classification task aims to assign a class label for each input example. It is similar to a binary classification task, but the assigned classes are drawn from K classes, where the value of K is higher than two. Formally, it can be defined as follows [4]:

Given a training data set of the form (x_i, y_i), where $x_i \in \Re^n$ is the i-th example, and $y_i \in \{1, 2, \ldots, K\}$ is the class label for the respective i-th example, a multi-class classification model, denoted by H, is defined as

$$\mathrm{H}(x_i) = y_i \tag{1}$$

in which H represents a model to predict a single label for a new unseen example.

Obtaining a good prediction for multi-class classification is challenging since the classification model has to distinguish between a large number of classes [5]. Therefore, many studies have been conducted on multi-class classification, and they commonly adopt one of three approaches: (1) apply some kind of transformation over the training data, (2) design a better learning algorithm by modifying or adjusting the hyper-parameters of existing algorithms, and (3) combination of both approaches.

A strategy that is widely adopted in text classification is to use different feature extraction techniques, such as term occurrence [6–9], term presence [8], and term frequency-inverse document frequency (TF-IDF) [8–12]. The following equations show examples of representations of the three types of extracted features in sequence, namely term occurrence ($\vec{d_{to}}$), term presence ($\vec{d_{tp}}$), and TF-IDF ($\vec{d_{tf-idf}}$).

$$\vec{d_{to}} = (2,\ 0,\ 3,\ 4,\ 0,\ 1,\ 5,\ 0) \tag{2}$$

$$\vec{d_{tp}} = (1,\ 0,\ 1,\ 1,\ 0,\ 1,\ 1,\ 0) \tag{3}$$

$$\vec{d_{tf-idf}} = (0.03,\ 0,\ 0.04,\ 0.06,\ 0,\ 0.01,\ 0.07,\ 0) \tag{4}$$

The examples of extracted features as shown in Equations (2)–(4) consist of 8 words, i.e., $(w_1, w_2, \ldots, w_8)$. Each element in the term occurrence feature shows how many words occur in a document. The example in Equation (2) shows that the document consists of 2 words of w_1, 3 words of w_3, 4 words of w_4, 1 word of w_6, and 5 words of w_7, whereas each element in the term presence feature shows whether the related word appears in a document. For example, Equation (3) means that the document consists of the words w_1, w_3, w_4, w_6, and w_7. Furthermore, in the TF-IDF feature, each value in the extracted feature shows the weight of how important the related word is in a document. For example, in Equation (4), the word importance weight of w_7 in the related document is 0.07.

Moreover, several studies on multi-class classification have been conducted in various fields, including those related to health. Research related to multi-class classification is mostly performed using classical machine learning methods, such as that of Yestigsen and Pratt [13], which uses the support vector machine (SVM) method to perform multi-class classification using a dataset in the form of a MEDLINE document from the OHSUMED dataset. In their study, the best classification results were obtained using a hybrid approach, with values of precision = 0.78, recall = 0.46, and F-measure = 0.60 [13].

The random forest method has also been used in some studies. For example, Bouazizi and Ohtsuki [14] used a Twitter dataset to obtain a classification accuracy of 60.2% for 7 different classes and 81.3% for binary classification. Further, in studies conducted by Ximale, Hinde, and Stone on multi-class classification of webpages on training courses, using the naive Bayes method, the highest F-measure obtained was 97% [15]. Another method that has been adopted for multi-class classification is logistic regression. For example, a study by Kurt et al. [16] used logistic regression to predict the presence of coronary artery disease (CAD) based on several factors, i.e., age, sex, family history of CAD, smoking status, diabetes mellitus, systemic hypertension, hypercholesterolemia, and body mass index (BMI). The outcomes of these studies suggest that the performances of the learning algorithms vary, and depend on the combination of various feature extraction techniques and learning algorithms. To identify the best performing algorithm requires a trial and error approach combining several feature extraction techniques and learning algorithms.

Therefore, in this study we perform an experimental benchmarking of the performance of four different algorithms that are widely implemented in multi-class classification problems, and specifically for the textual domain, based on classification of data with respect to stunting issues in Indonesia. In addition to the four algorithms, we also evaluate three types of textual features to obtain three different representation of the training data.

3. Methodology

This research adopts a multi-stage process, including data collection and selection, pre-processing, feature extraction, training, testing, and performance evaluation. We incorporate three types of extracted features—term occurrence (TO), term presence (TP), and term frequency-inverse document frequency (TF-IDF)—and four learning algorithms—logistic regression, naive Bayes, random forest, and support vector machine (SVM)—in the training process. Figure 1 provides an overview of the research methodology.

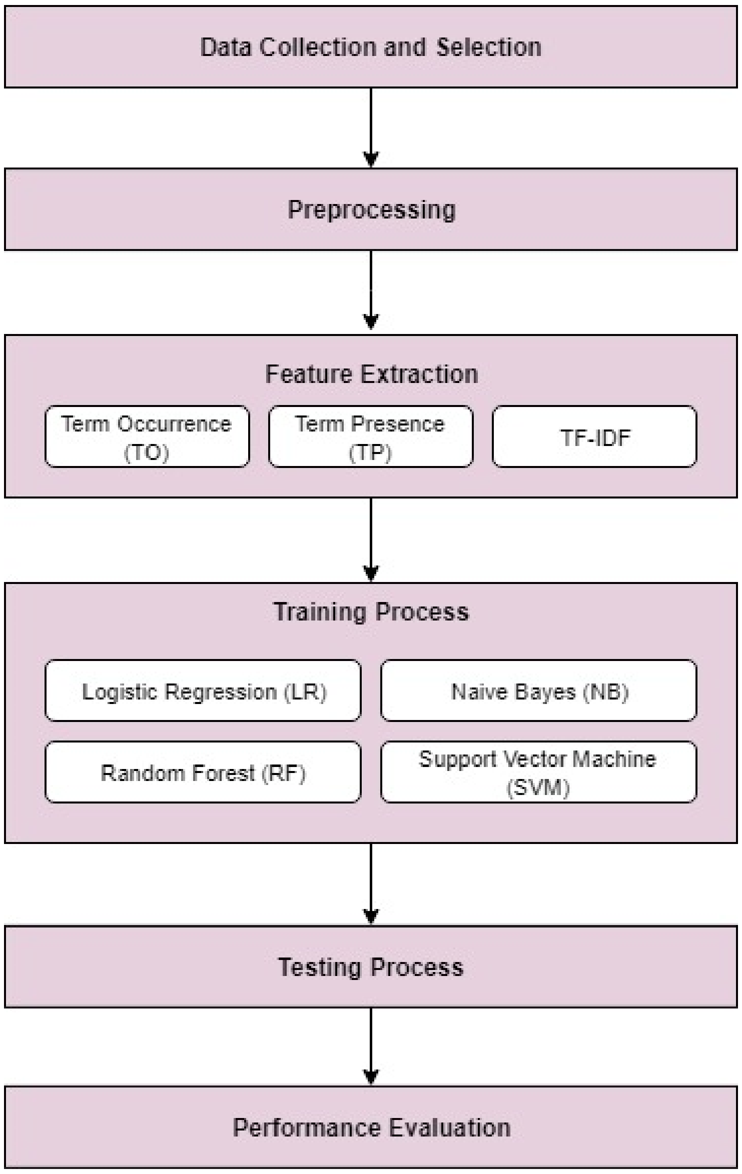

Figure 1. Research methodology.

3.1. Data Collection and Selection

The dataset is collected by exploring news portal websites, government websites, health websites, and retrieving Twitter data via Twitter API (Application Programming Interface). The data collected is in Indonesian and English. Specifically, data related to information about stunting is collected from news portal websites, government websites, and health websites. Data related to discussions among lay people is collected by crawling twitter posts (also called tweets) from the Twitter account @cegahstunting for data in Indonesian, and @1000days for data in English, while data unrelated to stunting is collected by crawling tweets from health, social, and political sources. Those three sources were selected since those sources often contain words or terms that are also mentioned in discussions or information about stunting. Therefore, it is expected that the same words/terms within different contexts will be differentiated into different classes. Details of the data obtained are provided in Table 1. Subsequently, we perform a round of random selection over the collected data to obtain the same number of documents for each class (stunting information, lay discussion on stunting, and unrelated to stunting) in our dataset. Hence, the data selection stage aims to create a balanced dataset.

Table 1. Collected data.

Data Source	Class Label [a]	Number of Documents	Data ID	Selected Data
Website portal in Indonesian Language	SI	200	D1	200
Website portal in English Language	SI	100	D2	100
Subtotal of Class Label SI				300
Twitter account @cegahstunting	LD	200	D3	200
Twitter account @1000Days	LD	2000	D4	100
Subtotal of Class Label LD				300
Twitter account @HealthNewsID	UN	300	D5	100
Twitter account @kpp_pa	UN	2000	D6	25
Twitter account @kemaritiman	UN	1996	D7	25
Twitter account @KementerianESDM	UN	1987	D8	25
Twitter account @Kemenhumham_RI	UN	2000	D9	25
Twitter account @HarvardChanSPH	UN	1996	D10	50
Twitter account @NatGeo	UN	2000	D11	50
Subtotal of Class Label UN				300

[a] SI—Stunting Information, LD—Lay Discussion on Stunting, UN—Unrelated to Stunting.

3.2. *Pre-Processing*

The pre-processing stage consists of three sequential processes, as illustrated in Figure 2.

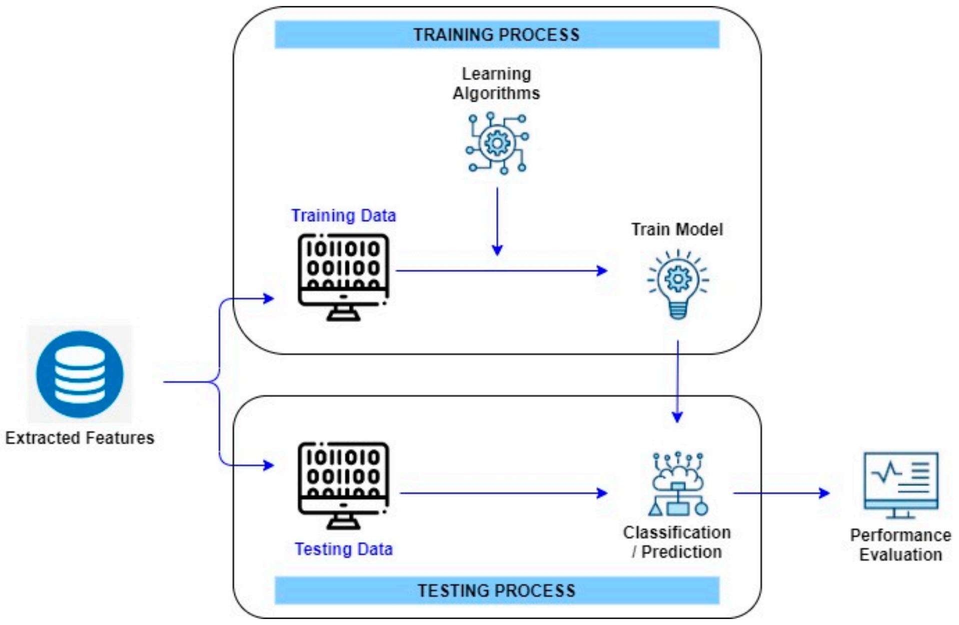

Figure 2. Illustration for Training and Testing Process.

1. Tokenization Tokenization is the process of decomposing a document into smaller units such as words, phrases, symbols, and other meaningful elements to map the words in the document [17]. These units are commonly referred to as tokens.
2. Stop words removal The stop words removal process filters out common words in documents, usually in the form of articles, prepositions, pronouns, or other words that do not add any significant meaning to the documents [10].
3. Stemming Stemming provides a mapping of different morphological variants of words into their base/common word (stem) [18]. Since we incorporate documents in both Indonesian and English, we adopt different stemming algorithms for the two languages. We used Porter stemmer for English documents, since it is efficient and simple [19], and Sastrawi stemmer for Indonesian documents. Sastrawi Stemmer is a modified stemmer algorithm from previously developed stemmers [20–22].

3.3. Feature Extraction

Three features are incorporated in this study, i.e., term occurrence (TO), term presence (TP), and term frequency-inverse document frequency (TF-IDF). TO, also known as raw term frequency, is a vocabulary vector, each of whose elements represents the number of occurrences of the corresponding word in a document. Many studies also refer to the TO feature as Bag-of-Words (BoW). TP is a simple form of TO in which each element of the vector is either 0 or 1, depending on whether the corresponding word occurs in the document or not.

TF-IDF is a numerical statistic that is intended to reflect the importance of a word to a document in a collection or corpus [23] so that the relevance of a word to all documents in the corpus can be known. The weight of a term can be calculated as follows [24]:

$$tf_{i,j} = \begin{cases} 1 + \log_2 f_{i,j}\ ,\ if\ f_{i,j} > 0 \\ 0,\ \text{otherwise} \end{cases} \tag{5}$$

$$idf_i = \log_2 \frac{N}{n_i} \tag{6}$$

$$w_{ij} = \begin{cases} 1 + \log_2 f_{i,j} \times \log_2 \frac{N}{n_i}\ ,\ if\ f_{i,j} > 0 \\ 0,\ \text{otherwise} \end{cases} \tag{7}$$

where

$tf_{i,j}$: term frequency weight of the index term k_i in the document d_j
$f_{i,j}$: frequency of occurrence of index term k_i in the document d_j
idf_j: inverse document frequency of term k_i
N: total number of documents in the corpus
n_i: number of documents containing the term k_i
$w_{i,j}$: tf-idf weighting scheme for term k_i in the document d_j

3.4. Training Process

In the training process, we implemented four different learning algorithms as mentioned in Figure 1, i.e., logistic regression, naive Bayes, random forest, and SVM. Figure 3 shows the illustration of both training dan testing processes and the following sub-sections explain each algorithm in detail.

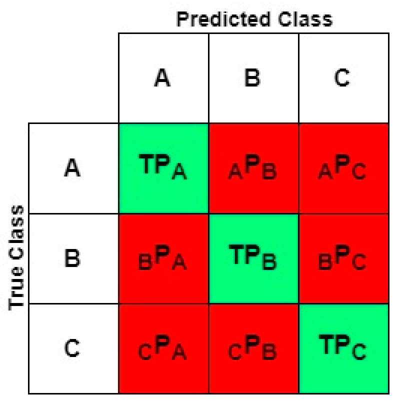

Figure 3. Confusion matrix.

3.4.1. Training Process Using Logistic Regression

Logistic regression is a machine learning method that adopts sigmoid curves. It is suitable for binary classification, but can be adapted for multiclass classification by using the one vs. rest scheme. The following is the sigmoid equation used in logistic regression:

$$f(x) = \frac{1}{(1+e^{-x})} \tag{8}$$

where

$f(x)$: sigmoid function of x

e: epsilon (2.7182)

x: input value

3.4.2. Training Process Using Naïve Bayes

The naive Bayes model determines the class of data by summing up the frequency of combinations of given dataset values to calculate a set of probabilities [25]. In classifying a sample of data, the naïve Bayes classifier assumes that the presence or absence of a particular feature of a class is independent of the presence or absence of other features [26]. The following is the equation of the naive Bayes classifier:

$$P\left(c\middle|X\vec{x}\right) = \frac{P(\vec{x}|c).\,P(c)}{P\left(\vec{x}\right)} \tag{9}$$

where

$\vec{x}$: a document with unknown class, $\vec{x} = (w_1, w_2, \ldots, w_V)$

c: document hypothesis, is a specific class, $c \in C$

C: set of classes

V: size of vocabulary

$P\left(c\middle|\vec{x}\right)$: posterior probability, probability that a document $\vec{x}$ belongs to the class c

$P(c)$: prior probability that any random document from the corpus belongs to class c

$P\left(\vec{x}\middle|c\right)$: likelihood probability, probability that a randomly chosen document from documents in the class c is the document $\vec{x}$

$P\left(\vec{x}\right)$: evidence probability, probability that a randomly chosen document from the corpus is document $\vec{x}$

There are two types of Naïve Bayes which is implemented in this study, i.e., multinomial naïve Bayes and Gaussian naïve Bayes. The Gaussian naïve Bayes was chosen for the TF-IDF feature because the values from TF-IDF are included in the continuous data category and there is a typical assumption that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. Meanwhile, the selection of multinomial naive Bayes for the term occurrence and term presence features is because the vectors of these two features represent the frequency and appearance of words that can be associated as a multinomial distribution. The prior probability can be estimated by dividing number of documents belonging to class c by the total number of documents. As likelihood probability for multinomial naïve Bayes can be calculated as follows:

$$P\left(\vec{x}\middle|c\right) = \prod_{i=1}^{V} P(w_i|c) = \prod_{i=1}^{V} \frac{count(w_i, c) + 1}{count(c) + V} \tag{10}$$

$$P\left(\vec{x}\right) = \sum_{c \in C} \left(P\left(\vec{x}\middle|c\right)P(c)\right) \tag{11}$$

where w_i: the i-th word in a document

i: index of word, $i = 1, 2, \ldots, V$

V: size of vocabulary

$P(w_i|c)$: the probability of the occurrence of the i-th word if the class is known to be c

$count(w_i, c)$: the number of occurrences of term w_i in training documents from class c, including multiple occurrences of a term in a document.

$count(c)$: the number of terms belonging to class c

Furthermore, the difference between Gaussian naive Bayes and multinomial naive Bayes lies in the calculation of the likelihood value as follows:

$$P\left(\vec{x}\middle|c\right) = \prod_{i=1}^{V} \mathcal{N}\left(w_i\middle|\mu_{c,i}, \sigma_{c,i}^2\right) = \prod_{i=1}^{V} \frac{1}{\sqrt{2\pi\sigma_{c,i}^2}} exp\left(-\frac{(w_i - \mu_{c,i})^2}{2\sigma_{c,i}^2}\right) \tag{12}$$

where

$\mu_{c,i}$: the mean of the feature values of term w_i associated with class c

$\sigma_{c,i}^2$: the variance of the feature values of term w_i associated with class c

3.4.3. Training Process Using Random Forest

The random forest method uses a large number of decision trees that operate as ensembles. Each tree casts a vote for the predicted class based on a random process, and the class with the most votes is the prediction of the model [27]. While simple in concept, the random forest method can be effective because the individual trees do not strongly correlate and therefore cancel out each other's errors, thereby potentially outperforming other individual models. To produce desirable outcomes, the random forest method requires two conditions, (1) there must be an actual signal in the feature so that the model built uses better features than random guessing, and (2) the predictions of the trees must have a low correlation with each other.

3.4.4. Training Process Using SVM

SVM is a method to compute the best hyperplane that separates two distinct classes [28]. SVM was originally designed for linear classification, but can now be used for non-linear classification problems by applying the kernel. In this study, we implemented non-linear SVM with RBF (Radial Basis Function) as a kernel function since it can be generally used for all types of data. In addition, we also employed some additional kernel functions, i.e., polynomial degree 3, linear, and sigmoid. We choose polynomial and linear kernel function since these kernels are less time consuming than the RBF. Thus, it is necessary to make a comparison to find out whether it can produce the same accuracy value or close to RBF, whereas the sigmoid kernel function comes from the Neural Networks field. Performing sigmoid kernel function in SVM is equivalent to a two-layer perceptron in which it is performed well in practice.

3.5. Testing Process

We implemented 10-fold cross validation to divide the dataset into two subsets of training and testing data. Therefore, the proportion of data splitting between training data and testing data was 90% of training data and 10% of testing data. Since we had 900 data samples, each fold consisted of 90 data samples, with 30 data samples from the stunting information class, 30 data samples from the lay discussions on stunting class, and 30 data samples from the unrelated to stunting class. As mentioned in Table 1, we employed a balanced dataset since the classification model is built based on imbalanced datasets that have minimal sensitivity to minority classes and can lead to poor classifier performance. We ran both the training and testing processes 10 times (for 10-fold cross validation), where each iteration was carried out by running the training process for 9 folds and the testing process for the remaining 1 fold. The iteration process was carried out so that each fold was tested.

3.6. Performance Evaluation

The performance of each learning algorithm was evaluated using the accuracy value, since we used a balanced dataset, i.e., 900 data samples across the three classes with 300 data samples for each class. In addition, there was no difference in the level of importance between classes so that evaluation metrics such as precision, recall, or F1-Measure were not required. As we had 3 classes in our multi-class classification task, i.e., stunting information class (A), lay discussion on stunting class (B), and unrelated to stunting class (C), we can compute the confusion matrix as shown in Figure 3.

The green cells in the figure represent true positives for each class, which are correctly classified samples. For example, TPA denotes the number of true positives for class A, i.e., the number of testing data samples that actually belong to class A and are also predicted as class A by the algorithm. By contrast, the red cells represent misclassified samples. For example, ${}_{A}P_{B}$ represents the number of samples with true class A, but predicted as B by the algorithm. The sum of ${}_{A}P_{B}$ and ${}_{A}P_{C}$ is the number of false negatives for class A. The sum of correctly classified samples and misclassified samples is equal to the size of the dataset. Finally, the accuracy value is the ratio between the number of correctly classified samples (all green blocks) and the total size of the dataset (all blocks). Since we implemented 10-fold cross validation, as previously explained, we computed the mean and standard deviation values of the accuracy obtained in experiment of each fold.

4. Results and Analysis

In this section we provide details of the experimental setup, as well as the experimental results and analysis.

4.1. Experimental Setup

As explained previously, the research data consisted of 900 data samples for three classes. The setup was implemented in the Python programming language and run in Google Collaboratory. We performed parameter tuning for each algorithm as follows:

1. Logistic regression: regularization parameter (C). There are 5 tested values of C, i.e., 0.01, 0.1, 1, 10 and 100.
2. Random forest: maximum number of features (auto, sqrt and 0.2) and number of trees to build before taking the maximum voting or averages of predictions (10, 50 and 100).
3. SVM: kernel function (RBF, Linear, Polynomial degree 3 and Sigmoid)

In this study, parameters for the naive Bayes were not tuned but used 2 different models as described in Section 3.4.2 and the justification for selecting the kernel functions that are tuned have been described in Section 3.4.4. Furthermore, we set the regularization parameter in the logistic regression method to improve generalization performance, namely the performance when classifying new data. We employed three options for maximum number of features random forest as offered by Python in the Scikit-learn library, whereas the number of trees to build before taking the maximum voting or averages of predictions was determined by choosing any value but still paying attention to the processor capability since a higher number of trees gives better performance but makes the code slower.

4.2. Experimental Results and Analysis

We report the accuracy values for each parameter value and each feature. The all result tables show the mean accuracy values from ten folds and the standard deviations of the accuracy from the same ten folds. For example, the value in the first row of the first column in Table 1, namely 0.60 ± 0.27, means that the experimental results using the Logistic Regression method with a regularization parameter of 0.01 and the employed feature of term occurrence has an average accuracy value of 60% with a standard deviation of 27%.

According to Table 2, the increase in regularization value is directly proportional to the increase in the accuracy value. Therefore, the best accuracy value for the three features is obtained at a regularization value of 100 (marked in yellow). The value of the regularization parameter is equal to 1 per λ, in which the λ value is used to control the complexity of the built model. If the lambda value is large, it will cause the model to be built very simply and this causes underfit. Subsequently, since the parameter regularization value is inversely proportional to lambda value, then increasing the regularization value is directly proportional to the increase of accuracy value.

Table 2. Result of parameter tuning for logistic regression.

Regularization Parameter Value	Features		
	TO	TP	TF-IDF
0.01	0.60 ± 0.27	0.55 ± 0.29	0.03 ± 0.06
0.1	0.74 ± 0.26	0.73 ± 0.25	0.74 ± 0.26
1	0.76 ± 0.22	0.78 ± 0.24	0.76 ± 0.22
10	0.78 ± 0.21	0.79 ± 0.22	0.78 ± 0.21
100	0.78 ± 0.20	0.79 ± 0.22	0.78 ± 0.20

According to Table 3, the best accuracy for all features was obtained for the following parameters, i.e., (i) maximum number of features is 0.2, and (ii) number of trees to build is 50. When we use the maximum number of features is 0.2, it means that we are increasing the employed features compared to the sqrt. This increases the accuracy value since at each node we have a higher number of options to be considered. While the second parameter—that is, the number of trees to build before taking the maximum voting or averages of predictions—can be determined based on trial and error. In this study, its best value is 50.

Table 3. Result of parameter tuning for random forest.

Maximum Number of Features	Number of Trees to Build	Features		
		TO	TP	TF-IDF
auto	10	0.72 ± 0.25	0.67 ± 0.26	0.67 ± 0.23
	50	0.69 ± 0.25	0.71 ± 0.26	0.73± 0.25
	100	0.72 ± 0.26	0.71 ± 0.27	0.73 ± 0.26
sqrt	10	0.70 ± 0.25	0.68 ± 0.26	0.66 ± 0.25
	50	0.71 ± 0.26	0.72 ± 0.26	0.72 ± 0.25
	100	0.72 ± 0.25	0.71 ± 0.27	0.72 ± 0.26
0.2	10	0.73 ± 0.22	0.73 ± 0.20	0.73 ± 0.21
	50	0.76 ± 0.21	0.75 ± 0.20	0.73 ± 0.21
	100	0.75 ± 0.21	0.75 ± 0.22	0.73 ± 0.21

The employed parameter tuning for SVM is kernel functions. As mentioned in Table 4, we observed four different kernel functions. The best kernel function for TF-IDF is RBF, while the best kernel function for TO and TP is linear function. There is no parameter tuning for Naive Bayes since the determination of the likelihood distribution must be in accordance with the employed features.

Table 4. Result of parameter tuning for support vector machine (SVM).

Kernel Functions	Features		
	TO	TP	TF-IDF
RBF	0.94 ± 0.05	0.94 ± 0.06	0.98 ± 0.03
Linear	0.97 ± 0.03	0.98 ± 0.03	0.97 ± 0.03
Polynomial (Degree 3)	0.63 ± 0.24	0.67 ± 0.25	0.79 ± 0.25
Sigmoid	0.92 ± 0.04	0.94 ± 0.04	0.95 ± 0.06

Based on the results mentioned in Tables 2–4, the summary of best accuracy for each feature is provided in Table 5. These values were computed for all pairs of learning algorithms and employed features.

Table 5. Experimental results in which cells in yellow represent the best performing pair of learning algorithms and features.

Learning Algorithm	**Features**			**Average**
	TO	**TP**	**TF-IDF**	
Logistic Regression	0.78 ± 0.20	0.79 ± 0.22	0.78 ± 0.20	0.78 ± 0.21
Naive Bayes	0.63 ± 0.24	0.62 ± 0.25	0.57 ± 0.26	0.61 ± 0.25
Random Forest	0.76 ± 0.21	0.75 ± 0.20	0.73 ± 0.21	0.75 ± 0.21
SVM	0.97 ± 0.03	0.98 ± 0.03	0.98 ± 0.03	0.98 ± 0.03
Average	0.79 ± 0.17	0.79 ± 0.18	0.77 ± 0.18	

According to Table 2, the best accuracy value is 0.98, with a standard deviation of 0.03, and it is obtained with SVM as the learning algorithm and TF-IDF as the employed feature. TF-IDF conceptually assigns a lower weight to a term that occurs in almost every document. Therefore, TF-IDF represents a term's relative importance within a document. As this study aims to classify documents regarding stunting information (class A), documents on lay discussions about stunting (class B), and documents that are not related to stunting (class C), the distribution of words with low TF-IDF in class A documents, for example, is very different from the distribution of the same words for documents that belong to class B or C. Hence, the TF-IDF feature can be considered a representative feature to be paired with SVM and the RBF kernel, which is capable of handling nonlinear relationships between class labels and attributes [29]. SVM outperforms other algorithms such as naive Bayes, logistic regression, and random forest because SVM generalizes well on larger documents and datasets, and can automatically model the interaction between features [30].

The implementation of naïve Bayes for the TO and TP features uses the multinomial variant of naïve Bayes. The results indicate that it provides better accuracy than the implementation for TF-IDF with Gaussian naïve Bayes. This is perhaps because, in classification tasks, whether a word occurs or not seems to be more important than the frequency of occurrence of the word, which renders the multinomial distribution better suited for the task than the Gaussian distribution.

5. Conclusions

In this study, we performed a comparative evaluation of the performances of several algorithms in order to provide a benchmark for algorithms for multi-class classification, and specifically for documents related to stunting issues in the Indonesian context. The results indicate that the SVM method coupled with TF-IDF was able to fairly extensively identify documents containing information and discussions among lay people related to stunting, producing the highest accuracy value of 0.98, with a standard deviation of 0.03, due to its capability to automatically model the interaction between features. The results can enable researchers and policy makers to choose among various approaches to extensively retrieve relevant discussions to study community awareness and sentiment towards stunting.

Author Contributions: Conceptualization, R.K.; methodology, R.K. and T.A.I.; software, T.A.I. and S.R.J.; validation, R.K. and A.F.H.; formal analysis, R.K. and B.S.; investigation, R.K. and T.A.I.; resources, S.R.J., T.A.I. and A.F.H.; data curation, T.A.I. and A.F.H.; writing—original draft preparation, R.K., S.R.J. and A.F.H.; writing—review and editing, R.K. and K.K.; visualization, T.A.I. and S.R.J.; supervision, R.K.; project administration, S.R.J. and K.K.; funding acquisition, R.K., K.K. and B.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Universitas Diponegoro under Grant 329-117/UN7.6.1/PP/2020 for fiscal year 2020.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. World Health Organization. *Global Nutrition Targets 2025: Stunting Policy Brief*; World Health Organization: Geneva, Switzerland, 2014.
2. Sapiie, M.A. Jokowi Wants Integrated Programs to Tackle Stunting, the Jakarta Post. 2018. Available online: http://www.thejakartapost.com/news/2018/04/05/jokowi-wants-integrated-program-to-tackle-stunting.html (accessed on 7 April 2018).
3. De Onis, M.; Branca, F. Childhood stunting: A global perspective. *Matern. Child Nutr.* **2016**, *12*, 12–26. [CrossRef] [PubMed]
4. Aly, M. *Survey on Multiclass Classification Methods*; Technical Report; California Institute of Technology: Padina, CA, USA, 2005.
5. Silva-Palacios, D.; Ferri, C.; Ramirez-Quintana, M.J. Improving Performance of Multiclass Classification by Inducing Class Hierarchies. *Proceedia Comput. Sci.* **2017**, *108*, 1692–1701. [CrossRef]
6. Momtazi, S.; Khudanpur, S.; Klakow, D. A Comparative Study of Word Co-occurrence for Term Clustering in Language Model-based Sentence Retrieval. In Proceedings of the Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Los Angeles, CA, USA, 2–4 June 2010; pp. 325–328.
7. George, S.K.; Joseph, S. Text Classification by Augmenting Bag of Words (BOW) Representation with Co-occurrence Feature. *IOSR J. Comput. Eng.* **2014**, *16*, 34–38. [CrossRef]
8. Satriaji, W.; Kusumaningrum, R. Effect of Synthetic Minority Oversampling Technique (SMOTE), Feature Representation, and Classification Algorithm on Imbalanced Sentiment Analysis. In Proceedings of the 2nd International Conference on Informatics and Computational Sciences, Semarang, Indonesia, 30–31 October 2018.
9. Kurniawan, S.; Kusumaningrum, R. Hierarchical Sentence Sentiment Analysis of Hotel Reviews using The Naive Bayes Classifier. In Proceedings of the 2nd International Conference on Informatics and Computational Sciences, Semarang, Indonesia, 30–31 October 2018.
10. Vijayarani, S.; Ilamathi, J.; Nithya, M. Preprocessing Techniques for Text Mining-An Overview. *Int. J. Comput. Sci. Commun. Netw.* **2015**, *5*, 7–16.
11. Campr, M.; Jezek, K. Topic Models for Comparative Summarization. In *Text, Speech, and Dialogue, Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2013; Volume 8082, pp. 568–574.
12. Trstenjak, B.; Mikac, S.; Donko, D. KNN with TF-IDF Based Framework for Text Categorization. *Procedia Eng.* **2014**, *69*, 1356–1364. [CrossRef]
13. Yetisgen-Yildiz, M.; Pratt, W. The effect of feature representation on MEDLINE document classification. *AMIA Annu. Symp. Proc.* **2005**, *2005*, 849–853.
14. Bouazizi, M.; Ohtsuki, T. Multi-Class Sentiment Analysis on Twitter: Classification Performance and Challenges. *Big Data Min. Anal.* **2019**, *2*, 181–194. [CrossRef]
15. Xhemali, D.; Hinde, C.J.; Stone, R.G. Naïve Bayes vs. Decision Trees vs. Neural Networks in the Classification of Training Web Pages. *Int. J. Comput. Sci. Issues* **2009**, *4*, 16–23.
16. Kurt, I.; Ture, M.; Kurum, A.T. Comparing performances of logistic regression, classification and regression tree, and neural networks for predicting coronary artery disease. *Expert Syst. Appl.* **2008**, *34*, 366–374. [CrossRef]
17. Kannan, S.; Gurusamy, V. Preprocessing Techniques for Text Mining. In Proceedings of the RTRICS, Tamil Nadu, India, 9–10 October 2014; pp. 1–6.
18. Tala, F.Z. *A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia*; Universiteit van Amsterdam: Amsterdam, The Netherlands, 2003.
19. Singh, J.; Gupta, V. A systematic review of text stemming techniques. *Artif. Intell. Rev.* **2016**, *48*, 157–217. [CrossRef]
20. Asian, J. Effective Techniques for Indonesian Text Retrieval. Ph.D. Thesis, RMIT University, Melbourne, Australia, 2007.

21. Arifin, A.Z.; Mahendra, I.P.A.K.; Ciptaningtyas, H.T. Enhanced Confix Stripping Stemmer and Ants Algorithm for Classifying News Document in Indonesian Language. In Proceedings of the International Conference on Information & Communication Technology and Systems (ICTS); 2009. Available online: https://www.researchgate.net/publication/228416000_ENHANCED_CONFIX_STRIPPING_STEMMER_AND_ANTS_ALGORITHM_FOR_CLASSIFYING_NEWS_DOCUMENT_IN_INDONESIAN_LANGUAGE (accessed on 1 December 2020).
22. Tahitoe, A.D.; Purwitasari, D. Implementation of Modified Enhanced Confix Stripping Stemmer for Indonesian Language Using Corpus Based Stemming Method. Master's Thesis, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia, 2010. Available online: http://digilib.its.ac.id/ITS-Undergraduate-3100010041042/14255 (accessed on 27 September 2020).
23. Jones, K.S. A Statistical Interpretation of Term Specificity and its Application in Retrieval. *J. Doc.* **2004**, *60*, 493–502. [CrossRef]
24. Baeza-Yates, R.; Ribeiro-Neto, B. *Modern Information Retrieval*, 2nd ed.; Pearson Education Limited: Edinburgh Gate, UK, 2011; pp. 68–72.
25. Saleh, A. Implementasi Metode Klasifikasi Naïve Bayes dalam Memprediksi Besarnya Penggunaan Listrik Rumah Tangga. *Creat. Inf. Technol. J.* **2015**, *2*, 207–217. (In Indonesian)
26. Pattekari, S.A.; Parveen, A. Prediction System for Heart Disease Using Naive Bayes. *Int. J. Adv. Comput. Math. Sci.* **2012**, *3*, 290–294.
27. Wang, B.; Gao, L.; Juan, Z. Travel Mode Detection Using GPS Data and Socioeconomic Attributes Based on a Random Forest Classifier. *IEEE Trans Intell. Transp. Syst.* **2018**, *18*, 1547–1558. [CrossRef]
28. Nugroho, A.S.; Witarto, A.B.; Handoko, D. Application of Support Vector Machine in Bioinformatics. In Proceedings of the Indonesian Scientific Meeting in Central Japan, Gifu, Japan, 20 December2003.
29. Hsu, C.; Chang, C.; Lin, C. *A Practical Guide to Support. Vector Classification*; Technical Report; National Taiwan University: Taipei, Taiwan, 2016.
30. Jurafsky, D.; Martin, J.H. Classification: Naive Bayes, Logistic Regression, Sentiment. In *Speech and Language Processing*; Prentice Hall: Upper Saddle River, NJ, USA, 2015; pp. 1–28.